



0.10 — Configuring your compiler: Compiler extensions

👤 ALEX¹ 🕒 MAY 5, 2023

The C++ standard defines rules about how programs should behave in specific circumstances. And in most cases, compilers will follow these rules. However, many compilers implement their own changes to the language, often to enhance compatibility with other versions of the language (e.g. C99), or for historical reasons. These compiler-specific behaviors are called **compiler extensions**.

Writing a program that makes use of a compiler extension allows you to write programs that are incompatible with the C++ standard. Programs using non-standard extensions generally will not compile on other compilers (that don't support those same extensions), or if they do, they may not run correctly.

Frustratingly, compiler extensions are often enabled by default. This is particularly damaging for new learners, who may think some behavior that works is part of official C++ standard, when in fact their compiler is simply over-permissive.

Because compiler extensions are never necessary, and cause your programs to be non-compliant with C++ standards, we recommend turning compiler extensions off.

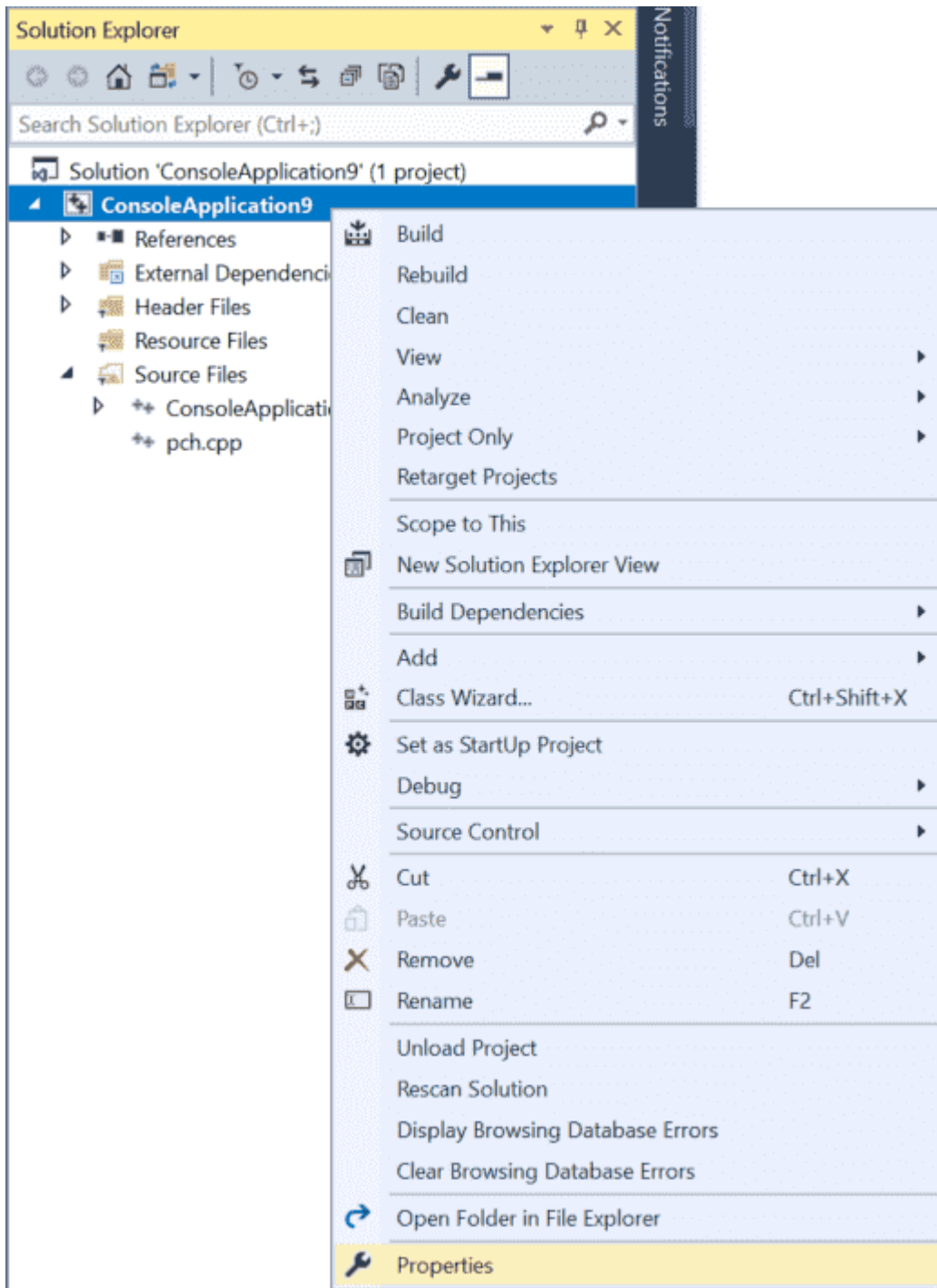
Best practice

Disable compiler extensions to ensure your programs (and coding practices) remain compliant with C++ standards and will work on any system.

Disabling compiler extensions

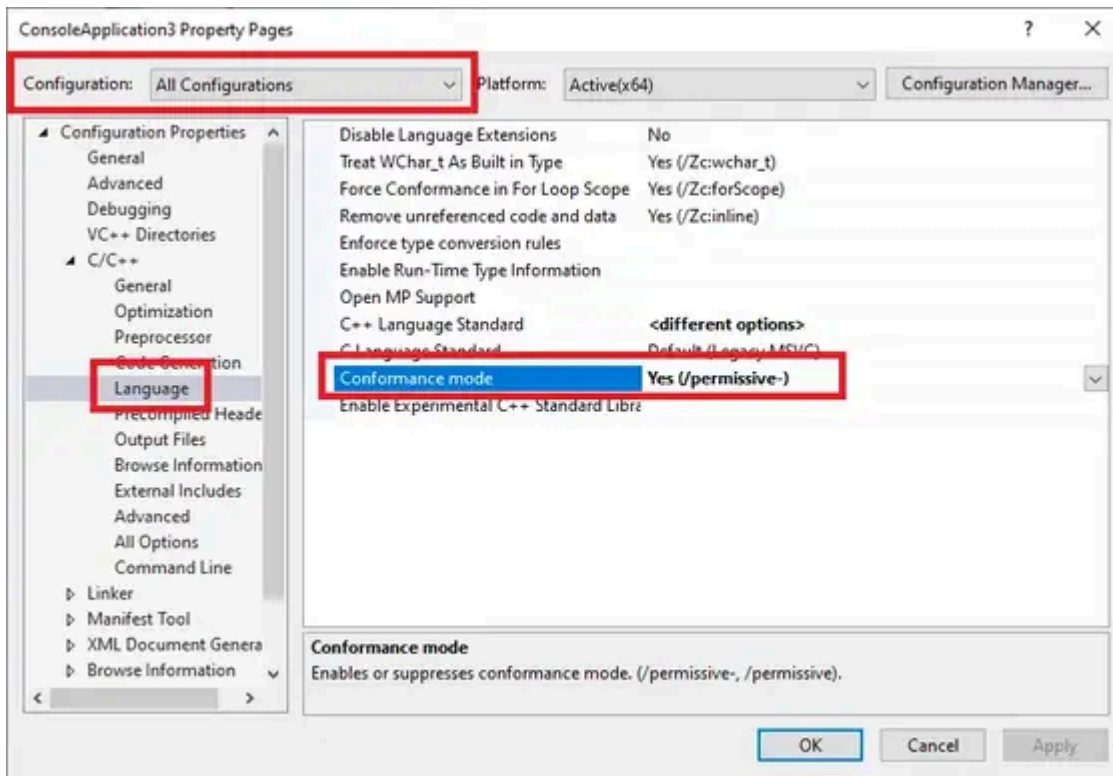
For Visual Studio users

To disable compiler extensions, right click on your project name in the *Solution Explorer* window, then choose *Properties*:



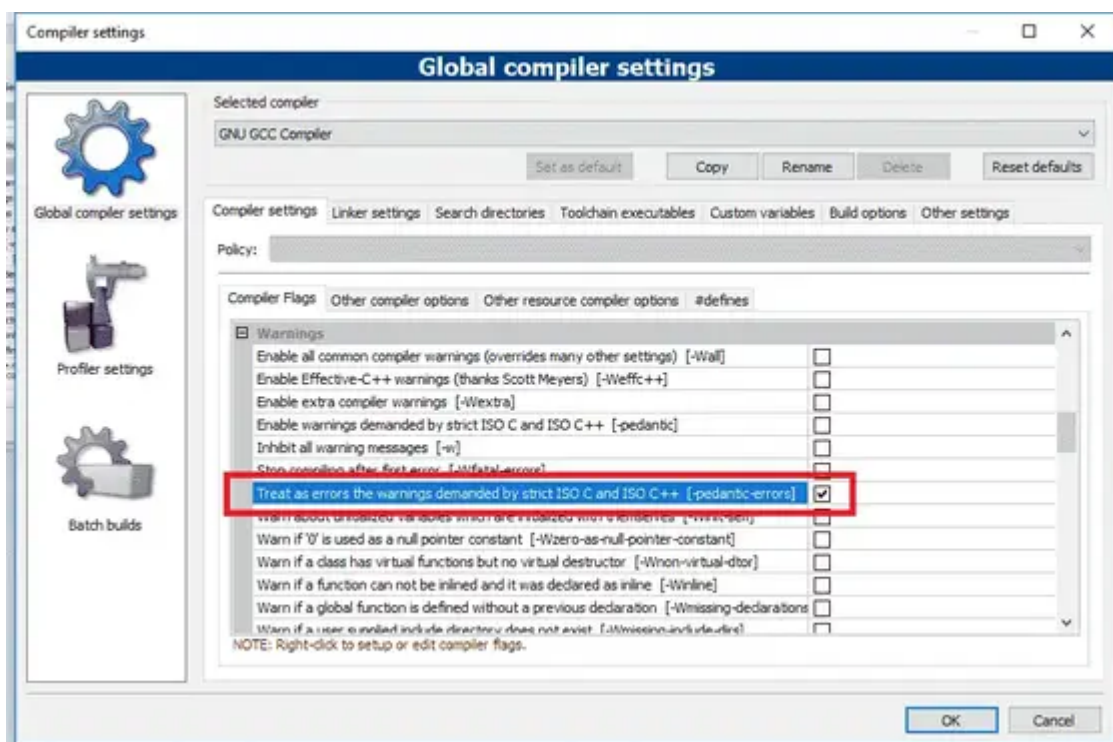
From the *Project* dialog, first make sure the *Configuration* field is set to *All Configurations*.

Then, click *C/C++ > Language tab*, and set *Conformance mode* to *Yes (/permissive-)* (if it is not already set to that by default).



For Code::Blocks users

Disable compiler extensions via *Settings menu > Compiler > Compiler flags tab*, then find and check the *-pedantic-errors* option.



For GCC/G++ users

You can disable compiler extensions by adding the *-pedantic-errors* flag to the compile command line.

For VS Code users

- Open the tasks.json file, find `"args"`, and then locate the line `"${file}"` within that section.
- Above the `"${file}"` line, add a new line containing the following commands:

```
"-pedantic-errors",
```

As of the time of writing, VS Code does not automatically add a newline to the end of code files that are missing it (something that is pedantically required by the C++ standard). Fortunately, we can tell VS Code to do so:

- Open VS Code and go to *File (Code if using a Mac) > Preferences > Settings*. This will open a settings dialog.
- Enter `insert final newline` into the search bar.
- In both the *Workspace Settings* and *User Settings* tabs, ensure the checkbox labeled *Files: Insert Final Newline* is checked.

Related content

Xcode users can refer to [Rory's comment](https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/comment-page-1/#comment-446983) (<https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/comment-page-1/#comment-446983>)², who kindly provided instructions.

A reminder

These settings are applied on a per-project basis. You need to set them every time you create a new project, or create a template project with those settings once and use that to create new projects.



[Next lesson](#)

0.11 [Configuring your compiler: Warning and error levels](#)

4



[Back to table of contents](#)

5



[Previous lesson](#)

0.9 [Configuring your compiler: Build configurations](#)

6



B **U** **URL** **INLINE CODE** **C++ CODE BLOCK** **HELP!**

Leave a comment...

Name*

Email*



Notify me about replies:



POST COMMENT

Find a mistake? Leave a comment above!

Avatars from <https://gravatar.com/>⁹ are connected to your provided email address.

167 COMMENTS

Newest ▼



Greg

April 9, 2024 9:56 am

Language extension is a big higher, first row, (its set to No), so why we enable conformance mode?

0

Reply



c noob

March 20, 2024 10:24 am

Please can someone explain me what is the main function? I have programmed in other languages before, and I thought that functions DO NOT AUTOMATICALLY EXECUTE.

0

Reply



Alex Author

Reply to [c noob](#)¹⁰ March 21, 2024 1:56 pm

main() is a special function that is automatically executed when the program starts.

We cover this in lesson <https://www.learncpp.com/cpp-tutorial/statements-and-the-structure-of-a-program/>

👍 0

➡ Reply



S.E.

🕒 January 24, 2024 3:36 am

Seems like there is little to do for Visual Studio 2022. Lets hope things continue to be so straight forward. Though admittedly, I do feel like I've missed something by moving on haha.

👍 3

➡ Reply



IceFloe

➡ Reply to [S.E.](#)¹¹ 🕒 February 7, 2024 5:45 pm

What did you mean by doing little? I think that for a beginner like me, this is kind of useful advice, although I don't know what extensions each compiler has, maybe they mean those standards, functions and libraries that are not established by the C++ standard, maybe in the future I will be able to create my own extensions, who knows))

👍 1

➡ Reply



S.E.

➡ Reply to [IceFloe](#)¹² 🕒 February 7, 2024 5:47 pm

I meant that there are few steps involved in the setup process when compared to the alternatives. Although its still a hassle.

👍 2

➡ Reply



IceFloe

➡ Reply to [S.E.](#)¹³ 🕒 February 7, 2024 5:56 pm

You are right in this regard, but it's better to worry about it now so that there are no headaches in the future, of course you can't foresee everything in advance, but you'll better understand and understand the compiler settings))

👍 0

➡ Reply



AbeerOrTwo

🕒 December 13, 2023 6:57 pm

Will do this once Xcode installs

👍 0

➡ Reply

**louisph**

🕒 August 28, 2023 1:23 pm

VSCode (on MacOS) task automatically add flag `-std=gnu++14`. As fix I override with `-std=c++20`.

My tasks.json:

```

1  {
2      "tasks": [
3          {
4              "type": "cppbuild",
5              "label": "C/C++: clang++ build active file",
6              "command": "/usr/bin/clang++",
7              "args": [
8                  "-fcolor-diagnostics",
9                  "-fansi-escape-codes",
10                 "-std=c++20",
11                 "-g",
12                 "-pedantic-errors",
13                 "${file}",
14                 "-o",
15                 "${fileDirname}/${fileBasenameNoExtension}.o"
16             ],
17             "options": {
18                 "cwd": "${fileDirname}"
19             },
20             "problemMatcher": [
21                 "$gcc"
22             ],
23             "group": "build",
24             "detail": "Task generated by Debugger."
25         }
26     ],
27     "version": "2.0.0"

```

and this is the executed terminal command: `/usr/bin/clang++ -std=gnu++14 -fcolor-diagnostics -fansi-escape-codes -std=c++20 -g gdb /Users/myusername/Code/cpp/main.cpp -o /Users/myusername/Code/cpp/main.o`

Why `-std=gnu++14` is added? should I remove this? How can I remove this?

Grazie mille :)



1



Reply

**marc**🗨️ Reply to [louisph](#) ¹⁴ 🕒 September 20, 2023 5:09 pm

same issue



0



Reply

**Hooman**

🕒 June 30, 2023 11:46 pm

For CLion, add the following to `CMakeLists.txt` file:

```
1 | set(CMAKE_CXX_STANDARD_REQUIRED ON)
2 | set(CMAKE_CXX_EXTENSIONS OFF)
```

👍 26

➡ Reply

**Young**🗨 Reply to [Hooman](#)¹⁵ 🕒 March 31, 2024 11:03 pm

Do I have to add these two lines in each CMakeLists.txt in each different project? Or is there a universal way to disable compiler extensions? Thanks

👍 0

➡ Reply

**kel**🗨 Reply to [Hooman](#)¹⁵ 🕒 July 1, 2023 1:39 am

Thank you!

👍 0

➡ Reply

**Yahya**

🕒 May 8, 2023 6:10 pm

For VS Code, there is no `"${file}"` under "args" in tasks.json file.

👍 1

➡ Reply

**MehdiAli**🗨 Reply to [Yahya](#)¹⁶ 🕒 July 5, 2023 6:31 am

From where do I open tasks.json file in VS Code?

👍 0



➡ Reply

**Sam**🗨 Reply to [MehdiAli](#)¹⁷ 🕒 July 28, 2023 11:01 am

Where you saved your .cpp file, there should be a folder called .vscode. If you open .vscode, there should be a file called tasks.json. Once you open tasks.json, located "args" (located right below "command"), and follow the guide.

 0 Reply**Hal** Reply to [Sam](#)¹⁸  December 9, 2023 1:37 am

I do not get a .vscode folder or a tasks.json file when i create/save cpp files.
How do I create one, is there a template I can follow?

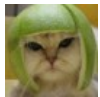
 0 Reply**rose** Reply to [Hal](#)¹⁹  February 11, 2024 3:50 am

you get it when you run the cpp file.

 0 Reply**Nico** May 3, 2023 7:38 am

VS Code -> For those of you still receiving the preALaunchTask error: no newline at end of file [-Werror,-Wnewline-eof] and have triple checked quotation on the "-pedantic-errors". You need to add an empty line at the end. So just press enter at the end of your code.

Hope this helps!

 0 Reply**Alex** Author Reply to [Nico](#)²⁰  May 5, 2023 10:19 pm

You can also tell VS Code to automatically add a newline if one is missing. I added instructions on how to do so to the lesson.

 3 Reply**Catherine** April 21, 2023 2:12 am

Can we have a clearer image on the Codeblocks version please

 0 Reply**samuel** Reply to [Catherine](#)²¹  June 6, 2023 4:13 pm

dont use code blocks that thing oldder and deprecated as shit



3



Reply

**I'm Batman**

🕒 April 12, 2023 10:39 pm

"-pedantic-errors", should be used instead of "-pedantic-errors", For VS Code users. The quotation mark symbol makes difference. "..." vs "..."



1



Reply

**Alex**

Author

💬 Reply to [I'm Batman](#)²² 🕒 April 13, 2023 9:13 pm

Oops, missed this comment somehow.

This is now fixed in the article.



2



Reply

Links

1. <https://www.learncpp.com/author/Alex/>
2. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/comment-page-1/#comment-446983>
3. <https://ikeamenu.com/humix/video/lpms8sWsckf>
4. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-warning-and-error-levels/>
5. <https://www.learncpp.com/>
6. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-build-configurations/>
7. <https://www.learncpp.com/configuring-your-compiler-compiler-extensions/>
8. <https://www.learncpp.com/cpp-tutorial/introduction-to-the-compiler-linker-and-libraries/>
9. <https://gravatar.com/>
10. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-594888>
11. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-592797>
12. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-593399>
13. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-593400>

14. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-586286>
15. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-582925>
16. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-580176>
17. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-583241>
18. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-584662>
19. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-590700>
20. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-580027>
21. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-579577>
22. <https://www.learncpp.com/cpp-tutorial/configuring-your-compiler-compiler-extensions/#comment-579240>